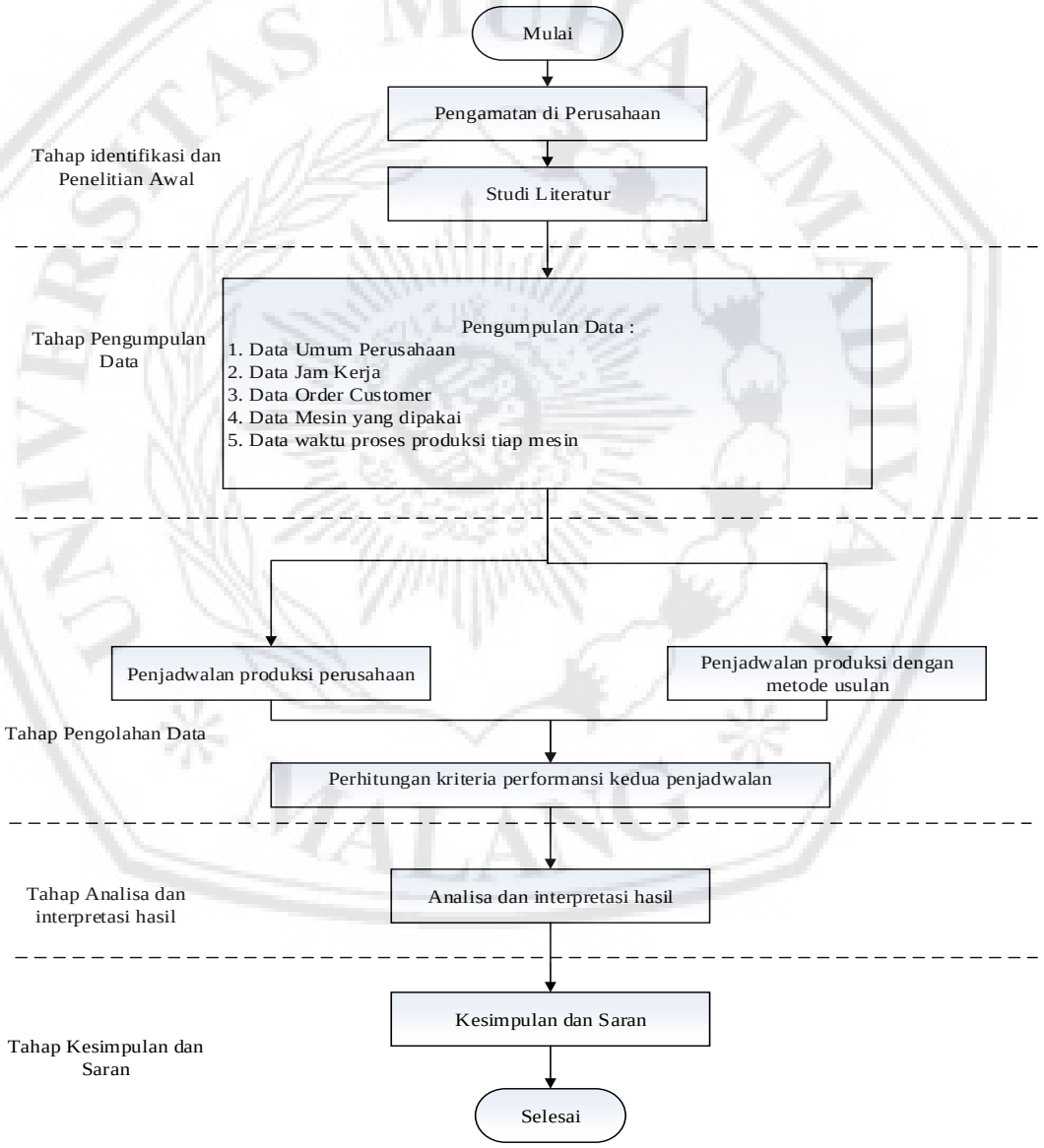
BAB III
METODOLOGI PENELITIAN

3.1 *Flowchart* Tahapan Penelitian

Flowchat atau diagram alir berguna memudahkan pembaca memahami metode penelitian yang dilakukan. Berikut adalah *flowchart* tahapan penelitian secara lengkap dapat dilihat pada gambar 3.1.



Gambar 3.1 *Flowchart* Metodologi Penelitian

3.2 Deskripsi Metodologi Penelitian

Berikut penjelasan bagaimana proses penelitian yang akan dilakukan mulai awal sampai selesai :

3.2.1 Tahap Identifikasi Dan Penelitian Awal

Tahap identifikasi dan penelitian awal adalah cara dari peneliti untuk dapat memperkirakan dan menguraikan apa yang sedang menjadi masalah pada perusahaan. identifikasi dan penelitian awal dalam penelitian ini terdapat beberapa tahap antara lain :

1. Pengamatan di Perusahaan

   Tahap identifikasi dan penelitian awal dimulai dari proses pengamatan di perusahaan dengan tujuan mengetahui kondisi awal perusahaan untuk dapat memberikan orientasi perusahaan kepada peneliti. Pada tahap ini peneliti melakukan penelitian di PT. Fronte Classic Indonesia, yaitu mempelajari serta memahami proses pembuatan produk yang berhubungan dengan penelitian. Serta melakukan pengidentifikasian dan perumusan masalah yaitu "Bagaimana penjadwalan produksi *flexible flowshop* di PT. Fronte Classic Indonesia dengan memodifikasi algoritma Srikandarajah & Sethi dengan Algoritma Hogdson untuk mengurangi jumlah job yang terlambat".
2. Studi literatur

   Tahap ini bertujuan untuk menunjang pencapaian tujuan pemecahan masalah dengan menggunakan pendekatan teori yang sesuai. Studi literatur yang digunakan mengacu dari berbagai sumber, baik dari buku, jurnal, maupun tugas akhir yang dijadikan referensi.

3.2.2 Tahap Pengumpulan Data

Tahap ini dilakukan untuk mengumpulkan data-data yang dibutuhkan untuk penelitian. Pengumpulan data dilakukan dengan wawancara kepada yang bersangkutan serta pengamatan secara langsung. Adapun data-data yang dibutuhkan peneliti antara lain :

1. Data umum perusahaan

   Data umum perusahan meliputi sejarah perusahaan, visi misi perusahaan, struktur organisasi, dll.

2. Data jam kerja

   Data yang berisi tentang jam kerja karyawan yang ada diperusahaan.

3. Data *order Customer*

   Data ini berisi tentang pesanan / *order* yang diterima perusahaan, seperti produk apa saja yang dipesan oleh *customer*, waktu diterima pesanan dan waktu pesanan harus dikirim ke konsumen (*due date*).

Table 3.1 Tabel Data *order customer*

Nama *Order*	Jumlah *Order*	Waktu pesanan	*Due Date*
Order 1			
Order 2			
Order n			

4. Data mesin yang dipakai

   Data mesin meliputi nama-nama dari mesin yang digunakan, jumlah mesin yang dipakai untuk menjalankan produksi untuk memenuhi pesanan atau order.

5. Data waktu proses produksi tiap mesin

   Data yang berisi data waktu proses produksi untuk mengerjakan pekerjaan pada tiap mesinnya.

3.2.3 Tahap Pengolahan Data

Pada tahap ini, peneliti akan melakukan pengolahan data yang sudah dikumpulkan sebelumnya. Data yang diperoleh dari perua sahaan akan dikelolah untuk menjadi inputan dari penjadwalan produksi itu sendiri. Berikut tahapan pengolahan data yang akan dilakukan :

1. Menghitung waktu tiap proses

   Tahap ini dilakukan perhitungan untuk mencari waktu tiap proses dengan cara mengkalikan waktu proses tiap mesin untuk 1 pcs atau 1 unit produk dengan jumlah unit order dari pelanggan. Dan dari sinilah inputan untuk melakukan penjadwalan (Matriks waktu proses pekerjaan).

2. Penjadwalan Produksi Pada Perusahaan

   Tahap melakukan penjadwalan *job* menggunakan penjadwalan yang saat ini diterapkan di perusahaan.

3. Penjadwalan Produksi Dengan Metode Usulan

   Penjadwalan produksi dengan metode usulan yaitu menjadwalkan produksi sesuai pesanan menggunakan modifikasi algoritma Srikandarajah & Sethi dan Algoritma Hogdson.

   Maksud dari modifikasi dalam penelitian ini adalah melakukan perubahan atau pergantian metode pada bagian yang kedua dan ketiga dalam algoritma srikandarajah & sethi. Awalnya pada bagian kedua dan ke tiga dalam algoritma Sriskandarajah & Sethi (1989) ialah menggunakan metode LPT dan mengunakan algoritma Johnson atau algoritma Gilmore dan Gomory. Sedangkan pada penelitian ini bagian kedua dan ketiga diganti dengan metode EDD dan Algoritma Hogdson. Hal ini dilakukan karena metode EDD dan Algoritma Hogdson dianggap bisa mengurangi permasalahan tentang minimasi jumlah *job* terlambat (Ho & Chang, 1995).

   Algoritma secara lengkap adalah sebagai berikut, Bagian pertama ialah membentuk kelompok *flowshop* mesin, masing-masing berisi satu mesin identik. Bagian kedua ialah bagian penetapan *Jobs* pada masing-masing kelompok *flowshop* yang sudah terbentuk sebelumnya, dalam bagian kedua ini peneliti menggunakan metode EDD. Bagian ketiga merupakan bagian pengurutan pekerjaan, dalam penelitian ini bagian ketiga menggunakan algoritma hogdson.

   Pada bagian Algoritma Hogdson, pada dasarnya algortima hogdson ini digunakan pada penjadwalan *single machine*. Dalam penelitian ini algortima hodgson digunakan untuk kasus *flowshop n-machine*. Maka dari itu ada beberapa bagian yang dimodifikasi oleh peneliti. Berikut adalah penjelasannya :

Table 3.2 Tabel perbedaan Algoritma Hogdson *single machine* dengan Algoritma Hodgson n-*machine*

Langkah	**Algoritma Hogdson *single machine*** (Sturm, 1970)	**Algoritma Hogdson pada penelitian ini (*flowshop n-machine*)**
1	Urutkan *Job* sesuai dengan due date yang paling kecil	Sama
2	Kemudian lakukan penjadwalan, jika tidak ada *job* yang terlambat maka urutan *job* tersebut sudah optimal. Jika ada yang terlambat lanjut ke langkah 3.	Sama
3	Jika *job i* pertama kali terlambat, maka cari *job a* sebelum *job i* yang mempunyai waktu pengerjaan paling lama. Kemudian hilangkan *job a* tersebut, untuk dikerjakaan setelah semua *job* tidak ada lagi yang terlambat setelah semuanya diproses.	Pada langkah yang ke tiga, pada dasarnya ialah sama. Sedikit berbeda pada bagaimana mencari waktu pengerjaan paling lama. Caranya adalah dengan menambah seluruh waktu seluruh proses operasi pada tiap job di tiap mesin. contoh : waktu pengerjaan *Job* a = M1 + M2 + ...+ Mn
4	Lakukan penjadwalan kembali seperti langkah 2 tanpa *job* yang terlambat sebelumnya, lakukan langkah 3 dan 4 hingga *job* tidak ada yang terlambat atau hanya menyisakan 1 *job* yang terlambat diurutan paling belakang.	Sama
5	Letakkan semua *job* yang dihilangkan karena keterlambatan tadi dalam urutan paling akhir di urutan penjadwalan yang sudah terbentuk.	Sama

Berikut ini adalah tabel perbedaan antara Algoritma Srikandarajah & Sethi dengan Algorima modifikasi Srikandarajah & Sethi dengan Algoritma Hogdson :

Table 3.3 Tabel perbedaan antara Algoritma Srikandarajah & Sethi dengan Algorima modifikasi Srikandarajah & Sethi dengan Algoritma Hogdson

Langkah	Algoritma Srikandarajah & Sethi	Langkah	Modikifikasi algoritma Srikandarajah & Sethi dengan Algoritma Hogdson
Bagian 1			
Bentuk kelompok mesin, masing-masing berisi mesin dari masing-masing mesin utama.			
1	Membentuk kelompok *flowshop* mesin, masing-masing berisi satu mesin dari masing-masing mesin utama. *F1, F2, ..., Fp.*	1	Membentuk kelompok *flowshop* mesin, masing-masing berisi satu mesin dari masing-masing mesin utama. *F1, F2, ..., Fp.*
2	Inisialisasikan waktu selesei dari *F1, F2, ..., Fp* = 0.	2	Inisialisasikan waktu selesei dari *F1, F2, ..., Fp* = 0.
Bagian 2			
Menetapkan pekerjaan untuk kelompok mesin. Menggunakan aturan LPT		**Menetapkan pekerjaan untuk kelompok mesin. Menggunakan aturan EDD**	
3	Untuk setiap pekerjaan Ji, $1 \leq i \leq n$, cari total waktu pengerjaan *tti = t 1i + t2i + ... + tmi.*	3	Untuk setiap pekerjaan Ji, $1 \leq i \leq n$, urutkan pekerjaan sesuai dengan *due date* terdekat hingga terlama. Terbentuklah *Job list.*
4	Mengurutkan pekerjaan dalam urutan waktu pemrosesan terbesar ke terkecil dari total waktu pengolahan tt*i*	4	Mencari waktu proses minimum *Fi* antara semua *flowshop (F1, F2, ..., Fp).*
5	Mencari waktu proses minimum *Fi* antara semua *flowshop*.	5	Menetapkan pekerjaan Jobi pertama dalam daftar Fi yang mempunyai waktu seleseinya paling kecil.
6	Menetapkan pekerjaan *Job* pertama dalam daftar *Fi* dan diurutkan.	6	Menambah total waktu tt*i* dari J*i* untuk total waktu yang dibutuhkan dari *flowshop (F1, F2, ..., Fp)* yang dipilih Fi, yaitu: *fi* = f*i + tt i*
7	Menambah total waktu tt*i* dari J*i* untuk total waktu yang dibutuhkan dari *flowshop* yang dipilih Fi, yaitu: *fi* = f*i + tt i.*	7	Hapus pekerjaan J*i* dari *Job list.*
8	Hapus pekerjaan J*i* dari daftar pekerjaan.	8	Ulangi langkah 4 sampai 7 sampai daftar pekerjaan kosong.
9	Ulangi langkah 4 sampai 7 sampai daftar pekerjaan kosong.		-
Bagian 3			
	Mengurutkan pekerjaan di setiap *flowshop (F1, F2, ..., Fp)*. Menggunakan algoritma Johnson		**Mengurutkan pekerjaan di setiap *flowshop (F1, F2, ..., Fp*). Menggunakan Algoritma Hogdson**
10	buat daftar waktu proses untuk seluruh pekerjaan, baik pada mesin 1 (M1) dan mesin terkhir (M2). Kemudian cari seluruh waktu proses untuk seluruh pekerjaan dan tentukanlah yang paling minimum.	9	Dalam masing-masing kelompok *flowshop,* Urutkan Job sesuai dengan *due date* yang paling kecil.

11	Jika waktu waktu proses yang paling minimum terdapat pada mesin pertama (misal), tempatkan job pada awal deret penjadwalan.	10	Kemudian lakukan perhitungan penjadwalan, jika tidak ada job yang terlambat maka urutan job tersebut sudah optimal. Jika ada yang terlambat lanjut ke langkah selanjutnya.
12	Jika waktu proses yang paling minimum ada pada mesin kedua (misal), tempatkan job pada akhir dari deret penjadwalan.	11	Jika *job i* pertama kali terlambat, maka cari *job a* sebelum *job i* yang mempunyai waktu pengerjaan paling lama. Kemudian hilangkan *job a* tersebut, untuk dikerjakaan setelah semua *job* tidak ada lagi yang terlambat setelah semuanya diproses.
13	Pindahkan *job-job* tersebut dari daftarnya dan susun dalam bentuk deret penjadwalan. Jika masih ada job yang tersisa ulangi kembali langkah 2, sebaliknya jika sudah tidak ada lagi *job* yang tersisa maka penjadwalan telah selesei.	12	Lakukan penjadwalan kembali seperti langkah 10 tanpa *job* yang terlambat sebelumnya
		13	Lakukan langkah 3 dan 4 hingga *job* tidak ada yang terlambat atau hanya menyisakan 1 *job* yang terlambat diurutan paling belakang.
		14	Letakkan semua *job* yang dihilangkan karena keterlambatan kedalam urutan paling akhir di urutan penjadwalan yang sudah terbentuk pada langkah 13.

Berikut ini adalah langkah-langkah algoritma secara keseluruhan setelah dimodifikasi :

1. **Bagian 1**

   Bentuk kelompok mesin, masing-masing berisi mesin dari masing-masing mesin utama.

   Langkah 1

   Membentuk kelompok *flowshop* mesin, masing-masing berisi satu mesin dari masing-masing mesin utama. $F_1, F_2, ..., F_p$.

   Langkah 2

   Inisialisasikan waktu selesei dari $F_1, F_2, ..., F_p = 0$.

2. **Bagian 2**

   Menetapkan pekerjaan untuk kelompok mesin.

   Langkah 3

   Untuk setiap pekerjaan Ji, $1 \leq i \leq n$, urutkan pekerjaan sesuai dengan *due date* terdekat hingga terlama. Terbentuklah *Job list.*

   Langkah 4

   Mencari waktu proses minimum F_i antara semua *flowshop* ($F_1, F_2, ..., F_p$).

   Langkah 5

   Menetapkan pekerjaan *Jobi* pertama dalam daftar *Fi* yang mempunyai waktu seleseinya paling kecil.

   Langkah 6

   Menambah total waktu tt*i* dari J*i* untuk total waktu yang dibutuhkan dari *flowshop* ($F_1, F_2, ..., F_p$) yang dipilih Fi, yaitu: $f_i = f_i + tt_i$.

   Langkah 7

   Hapus pekerjaan J_i dari *Job list*.

Langkah 8

Ulangi langkah 4 sampai 7 sampai daftar pekerjaan kosong.

Setelah langkah 8, pekerjaan ini terkelompok ke dalam kelompok *flowshop* dan dialokasikan ke pasangan kelompok mesin *flowshop* ($F_1, F_2, ..., F_p$).

3. **Bagian 3**

Mengurutkan pekerjaan di setiap *flowshop* ($F_1, F_2, ..., F_p$).

Langkah 9

Dalam masing-masing kelompok *flowshop*, Urutkan Job sesuai dengan *due date* yang paling kecil.

Langkah 10

Kemudian lakukan perhitungan penjadwalan, jika tidak ada job yang terlambat maka urutan job tersebut sudah optimal. Jika ada yang terlambat lanjut ke langkah selanjutnya.

Langkah 11

Jika *job i* pertama kali terlambat, maka cari *job a* sebelum atau sama dengan *job i* yang mempunyai waktu pengerjaan paling lama. Kemudian hilangkan *job a* tersebut, untuk dikerjakaan setelah semua *job* tidak ada lagi yang terlambat setelah semuanya diproses.

Langkah 12

Lakukan penjadwalan kembali seperti langkah 10 tanpa *job* yang terlambat sebelumnya

Langkah 13

Lakukan langkah 11 dan 12 hingga *job* tidak ada yang terlambat atau hanya menyisakan 1 *job* yang terlambat diurutan paling belakang.

Langkah 14

Letakkan semua *job* yang dihilangkan karena keterlambatan kedalam urutan paling akhir di urutan penjadwalan yang sudah terbentuk pada langkah 13.

Setelah Langkah 14, penjadwalan selesai dan penyelesaian keseluruhan waktu *makespan* telah ditemukan. Dan bisa dilihat jumlah job yang terlambat.

4. Perhitungan Kriteria Performansi Kedua Penjadwalan

   Pada tahap ini dilakukan perhitungan kriteria performansi dari kedua penjadwalan. Kriteria performansi yang digunakan adalah jumlah pekerjaan yang terlambat.

3.2.4 Analisa dan Interpretasi Hasil

Pada tahap analisa dan interpretasi hasil akan dilakukan analisa hasil perhitungan kriteria performansi minimasi jumlah pekerjaan yang terlambat dari penjadwalan perusahaan serta penjadwalan yang diusulkan.

3.2.5 Kesimpulan dan Saran

Tahap akhir dari penelitian ini adalah penarikan kesimpulan atas keseluruhan hasil yang diperoleh dari langkah-langkah penelitian yang dilakukan. Penarikan kesimpulan ini merupakan jawaban dari permasalahan yang ada. Selain itu juga akan diberkan saran sebagai masukan yang positif berkaitan dengan hasil penelitian.